О шифровальных и дешифровальных машинах

Часть 11

А.К.ДЬЮДНИ

«То, что зашифровано на одной машине, можно легко расшифровать на другой».

Из книги А.Ходжеса «Алан Тьюринг: Загадка».

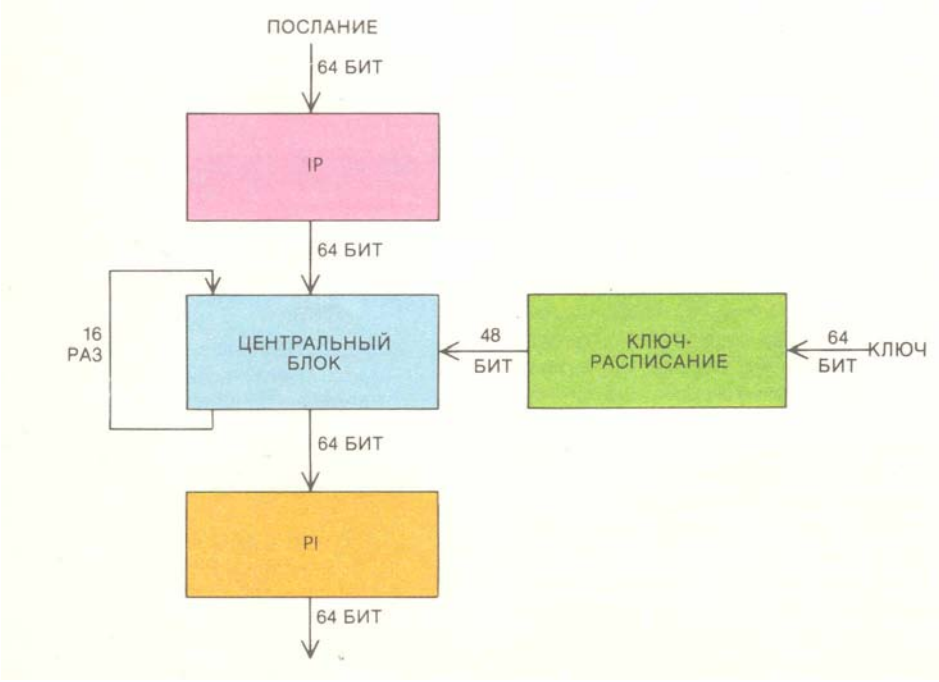
ДО ВТОРОЙ мировой войны засекречивание сообщений с помощью кодов и шифров представляло собой трудоемкую, утомительную ручную операцию. В военное время эту работу выполняли на электромеханических машинах, подобных немецкой «Enigma». В настоящее время шиФровкой текстов занимаются электронные компьютеры. Они быстро превращают сообщение на естественном языке в соответствии с секретным «ключом» в зашифрованный текст и передают его по телефонным или другим каналам электронной связи. Компьютеры, снабженные тем же самым ключом, также быстро расшифровывают полученное сообщение. Во всяком случае на такой основе строится Стандарт кодирования данных (скд) - метод шифрования информации, принятый Национальным бюро стандартов США в 1977 г. и широко применяемый сейчас в гражданских и, возможно, военных системах связи.

Ниже я приведу полное описание СКД, и это, по-видимому, будет первой широкой публикацией об этой системе. Я не рискую раскрыть каких-либо важных секретов, потому что содержание любой шифровки останется тайной, пока не станет известен ключ шифра. И все же в последнее время в науке о шифровании и дешифровании акцент делался на ключах, некоторые из которых общедоступны, другие предназначены лишь для личного пользования. Личный ключ скрывается при этом в хорошо известной задаче, которую никто не может решить.

Независимо от того, какая часть информации остается общедоступной, а какая засекречивается, любая компьютерная система, работающая с шифрами, основана на алгоритме кодирования и алгоритме декодирования. Алгоритм кодирования для скд первоначально был разработан в научно-исследовательском центре корпорации IBM. Подобно машине «Enigma», описание которой было приведено в предыдушем номере журнала, в системе скд используется последовательность операций кодирования, по отдельности очень простых, но в совокупности производящих сложное преобразование. В машине «Enigma» операции кодирования выполнялись с помощью механических колес; в системе скд они осушествляются либо программными модулями, либо микропроцессорами.

Поскольку кодирование выполняется на компьютере, в роли кодируемых символов выступают не буквы алфавита (как это было в машине «Enigma»), а биты, или двоичные разряды. В каждой операции кодирования по системе скд участвует лишь одна цепочка длиной 64 бит. Поэтому перед тем как применять алгоритм, нужно сначала перевести исходный текст сообщения в цепочки по 64 бит. Для этого можно применить практически любой способ. Например, поскольку пяти битов оказывается достаточно, чтобы сосчитать до 31 в двоичной системе счисления, можно выбрать двоичные представления чисел в диапазоне от 0 до 25 для 26 букв латинского алфавита. При такой схеме соответствия цепочка длиной 64 бит способна представлять 12 последовательных букв текста на англиЙском языке (при этом 4 бит останутся свободными).

Сообщение на исходном языке, шифруемое по системе СКД, можно представлять в виде реки, которая многократно делится на рукава, образуя сложнейшую речную систему. действительно, наилучший способ проследить за последовательностью отдельных операций - это воспользоваться «картой» реки, т. е. диаграммой алгоритма (см. рисунок слева). От дельные модули на диаграмме, подобно водопадам, перемешивают и вспенивают воду. Каждая цепочка исходного текста встречает модули трех различных типов на своем бурном пути к выходному каналу: перестановки, сдвиги влево и замены. Операции, выполняемые модулями каждого типа, удобно представить в виде таблиц.



Блок-схема системы кодирования данных eСКД

Таблица перестановок показывает, каким образом порядок битов в цепочке изменяется при воздействии данного модуля перестановки. Как пользоваться подобными таблицами, можно показать на простом примере. Предположим, что модуль перестановки воздействует на двоичную цепочку 1011 в соответствии со следующей (горизонтальной) таблицей: 3 1 2 4. Это означает, что третий бит цепочки переставляется на первую позицию, первый бит - на вторую позицию, второй бит - на третью, а последний остается на своем месте. Следовательно, применяя эту таблицу к числу 1011, мы получим число 1101.

В СКД используется несколько различных таблиц перестановок: так называемая первоначальная перестановочная таблица IP, таблица выбора бита E, или просто E, пара таблиц выбора-перестановки PC1 и PC2 и, наконец, обратная таблица перестановки P1. Эти таблицы приведены справа.

Элементы некоторых таблиц по длине не равны 64 бит, поскольку их входные и выходные цепочки содержат меньше 64 бит. Например, таблица выбора бита E принимает на входе цепочку длиной 32 бит и расширяет ее на выходе до цепочки длиной 48 бит. Не удивительно поэтому, что, глядя в соответствующую таблицу, мы обнаруживаем, что некоторые элементы встречаются в ней более одного раза. Модули PC1 и PC2 оказывают противоположное действие: они сжимают проходящие через них цепочки. Поэтому в соответствующих таблицах некоторые элементы отсутствуют.

Заинтересовавшимся тайнописью читателям, которые пожелают перенести эти таблицы, так же как и сам стандарт СКД, в свои домашние компьютеры, придется обойтись без подсказок с моей стороны. Поскольку мы ограничены размерами статьи, я лишь упомяну о том, что операторы присвоения, используемые при работе с таблицами перестановок, должны быть следующего вида:

$$T(k) \rightarrow M(P(k)),$$

где массив *M* содержит биты цепочки, над которой выполняется перестановка, массив *P* содержит таблицу, *T* - временный массив, а *k* отсчитывает новые позиции битов. Составленная таким образом компьютерная программа будет помещать элемент с номером *P(k)* массива *M* на *k*-ю позицию *T*. Когда все элементы *T* определены, *M* можно установить равным *T*.

Система СКД как бы запирает сообщения на замок при помощи секретного ключа - произвольной цепочки из 64 бит, которую пользователь системы задает в качестве входных данных модулю ключа-расписания на начальном этапе кодирования. Ключ можно мысленно представить как речку-приток шириной 64 бит, которая сужается до 56 бит в модуле PC1, делится на два 28-битных канала, снова объединяющихся в модуле PC2, и вливается в основную реку сообщения последовательными порциями в виде 16 48-битных притоков, по одному для каждого шага главного итерационного цикла системы СКД.

Находясь еще в модуле ключа-расписания, два 28-битных канала встречаются с другим течением, перемещающим их биты влево. Как можно видеть из верхней таблицы на с. 84, на первом, втором, девятом и 16-м шаге итераций происходит смешение битов двух цепочек влево на одну позицию. В остальных случаях цепочки сдвигаются влево на два бита.

Такие смешения влево легко реализуются в программе. 28 бит сегмента ключевой последовательности при: сваиваются, бит за битом, временному массиву таким образом, чтобы его i-й элемент принимал значение либо **(i +** 1)-го, либо **(i +** 2)-го бита сегмента. Конечно, последний элемент (или два) временного массива нужно установить равным первому биту (или двум) ключевой последовательности.

В систему СКД входят также восемь модулей замены, каждый из которых преобразует 6-битную цепочку в 4-битную. Биты числа, подвергающегося преобразованию, сами подсказывают, какими значениями индексов строки и столбца следует воспользоваться в соответствующей таблице: первый и последний биты определяют индекс строки, а средние четыре бита - индекс столбца (см. рисунок на с. 84). Например, 6-битное число 111010 дает индекс строки 10 (крайние биты) и индекс столбца 1101 (внутренние биты). Эти двоичные числа представляют соответственно десятичные числа 2 и 13. Элемент в таблице S1 в строке 2 и столбце 13 - это десятичное число 10. В двоичной форме число 10 выглядит как 1010, которое и появится на выходе модуля S1, если на вход ему поступила цепочка 111010. (Большинству шифровальщиков, работающих с персональным компьютером, возможно, потребуются еще две процедуры преобразования для того, чтобы переводить двоичные числа в десятичные и наоборот.)

Теперь рассмотрим, как взаимодействуют между собой эти различные потоки битов. Цепочка из 64 бит поступает в модуль IP и подвергается перестановке, описанной в соответствующей таблице. Затем она проходит в центральный блок, где основной кодирующий алгоритм перемешивает ее до неузнаваемости в своих многочисленных вихрях, прежде чем она дойдет до модуля P1, где она подвергается перестановке, обратной по отношению к той, которую выполняет модуль IP.

IP	E	PC1	PC2	P	PI	
58	32	57	14	16	40	1
50	1	49	17	7	8	2
42	2	41	11	20	48	3
34	3	33	24	21	16	4
26	4	25	1	29	56	5
18	5	17	5	12	24	6
10	4	9	3	28	64	7
2	5	1	28	17	32	8
60	6	58	15	1	39	9
52	7	50	6	15	7	10
44	8	42	21	23	47	11
36	9	34	10	26	15	12
28	8	26	23	5	55	13
20	9	18	19	18	23	14
12	10	10	12	31	63	15
4	11	2	4	10	31	16
62	12	59	26	2	38	17
54	13	51	8	8	6	18
46	12	43	16	24	46	19
38	13	35	7	14	14	20
30	14	27	27	32	54	21
22	15	19	20	27	22	22
14	16	11	13	3	62	23
6	17	3	2	9	30	24
64	16	60	41	19	37	25
56	17	52	52	13	5	26
48	18	44	31	30	45	27
40	19	36	37	6	13	28
32	20	63	47	22	53	29
24	21	55	55	11	21	30
16	20	47	30	4	61	31
8	21	39	40	25	29	32
57	22	31	51		36	33
49	23	23	45		4	34
41	24	15	33		44	35
33	25	7	48		12	36
25	24	62	44		52	37
17	25	54	49		20	38
9	26	46	39		60	39
1	27	38	56		28	40
59	28	30	34		35	41
51	29	22	53		3	42
43	28	14	46		43	43
35	29	6	42		11	44
27	30	61	50		51	45
19	31	53	36		19	46
11	32	45	29		59	47
3	1	37	32		27	48
61		29			34	49
53		21			2	50
45		13			42	51
37		5			10	52
29		28			50	53
21		20			18	54
13		12			58	55
5		4			26	56
63					33	57
55					1	58
47					41	59
39					9	60
31					49	61
23					17	62
15					57	63
7					25	64

Таблицы перестановок, применяемые в системе СКД

На вход центрального блока поступает не только результат преобразования модуля IP, но и 48-битные ключи (пронумерованные от 1 до 16), которые генерируются модулем ключа-расписания из заданной пользователем ключевой цепочки длиной 64 бит. Одни и те же основные операции выполняются в центральном блоке 16 раз, по одному разу для каждого из различных 48-битных ключей. Подробнее, 64-битная цепочка из модуля IP расшепляется на правую и левую половинки по 32 бита. Копия правой половинки сливается затем с ключом 1 втак называемом F-модуле. Результат складывается по модулю 2 с левой половинкой. (Сложение по модулю 2 подобно обычному сложению, за исключением того, что 1 плюс 1 равно 0.) После этого 32-битная сумма и исходная 32-битная половинка цепочки меняются ролями перед следуюшим шагом итераций (на этот раз с ключом 2), при этом сумма становится правой половинкой 64-битной цепочки, а правая половинка подставляется вместо левой. По завершении цикла из 16 таких итераций со слиянием и переменой мест две 32-битные цепочки снова объединяются, образуя единое целое - цепочку из 64 бит.

Операция F, пожалуй, стоит того, чтобы построить для нее отдельную диаграмму (см. рисунок на с. 86). Копия правой половинки текущей 64-битной цепочки входит в модуль выбора битов E, где она раСшИРЯется до 48 бит. Затем она складывается по модулю 2 с 48-битным ключом под номером, соответствующим данному шагу итераций. После этого 48 бит разветвляются сразу на 8 речушек, шириной по 6 бит каждая. Эти цепочки пробегают через модули замены от SI до S8, где каждая из них сжимается до четырехбитного числа. Поэтому, когда 8 речушек снова сливаются в единый поток, общая ширина цепочки уже равна только 32 бит. В этой точке модуль P снова перемешивает цепочку.

Если читателю хватило упорства внимательно проследить за всем этим описанием с начала до конца, то у него, естественно, может возникнуть вопрос, а возможно ли вообще восстановить исходный текст сообщения по шифровке после того «турбулентного перемешивания, которому подвергается этот текст, проходя сквозь систему СКД? Может быть, это настолько же трудно, как заставить воду течь в гору? В действительности дело обстоит именно так: прокрутим все основные операции центрального блока в обратном направлении. Направив 64-битную цепочку шифровки назад через модуль IP (аналогичная операция выполняется при кодировании сообщения), заставим цепочку повторить 16 шагов итераций в обратном порядке - от лежащего в основании центрального блока каскада со слиянием и переменой мест до вершины блока. Биты ключа также должны быть введены в модуль ключа-расписания в обратном порядке. После еще одной перестановки модулем PI исходное сообщение, подобно гейзеру, вырвется вверх.

Достаточно ли надежна система СКД на самом деле? Некоторые критики говорят, что нет. Еще до того, как система СКД была принята в качестве стандарта, У.Диффи и М.Е.Хеллман из Станфордского университета предупреждали, что СКД уязвим по отношению к лобовой (т. е. с исчерпывающим перебором вариантов) атаке, предпринятой достаточно хорошо оснащенным противником, скажем, способным построить параллельные компьютеры с быстродействием, которое позволяет перебирать 10 миллиардов возможных ключей в секунду.

Другое слабое звено системы СКД связано с необходимостью рассылать секретные ключи как отправителям, так и получателям сообщений. Это упрощает работу для агентов противника, у них появляется возможность каким-то образом раздобыть ключ и, пользуясь им, расшифровать сообщения. Поэтому 12 лет назад Диффи и Хеллман предложили рассылать так называемые общественные ключи по абонентам сети связи.

Схема распределения общественных ключей основана на очень большом простом числе p и числе основанияа. Число a специально подбирается таким образом, чтобы a^n по модулю p, где n принимает значения 0, 1,2, ... , давало все целые числа в диапазоне от 1 до p - 1. Каждому абоненту сети связи известны числар и a, и каждый абонент должен представить свой ключ, который будет указан в справочнике сети. Например, абонент i выбирает персональный «полу-

НОМЕР ИТЕРАЦИИ	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
ЧИСЛО СДВИГОВ ВЛЕВО	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

		КОЛОНКА															
	РЯД	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
S1	0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
	1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
	2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
	3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13
S2	0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
	1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
	2	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
	3	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9
S3	0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
	1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
	2	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
	3	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12
S4	0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
	1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
	2	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
	3	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14
S5	0	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
	1	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
	2	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
	3	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3
S6	0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
	1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
	2	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
	3	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13
S7	0	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
	1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
	2	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
	3	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12
S8	0	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
	1	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
	2	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
	3	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

Таблица сдвига влево (вверху) *и таблицы замены* (внизу)

ключ» xi и запирает его в сейфе. Затем он вычисляет числоki по формуле

$$k_i = a^{x_i} (\mathrm{mod}\, p)$$

и высылает его администрации сети с тем, чтобы оно было опубликовано в справочнике под его именем. То же самое делают все абоненты.

Если теперь абонент *j* пожелает передать секретное сообщение абоненту i, он найдет в справочнике число k_i и, воспользовавшись своим личным полуключом x_j, вычислит комбинированный ключ *kij,* возведя k_i в степень x_j по модулю *p.* Другими словами,

$$k_{ij} = k_i^{x_i} (\mathrm{mod}\, p).$$

Зная, что сообщение послал абонент *j,* абонент **i** аналогично вычисляет ключ k_{ji}, возводя k_j в степень x_i по модулю *p.* Читателям, возможно, будет интересно убедиться, что оба абонента на самом деле пользуются одним и тем же ключом, другими словами, числа k_{ij} и k_{ji} равны.

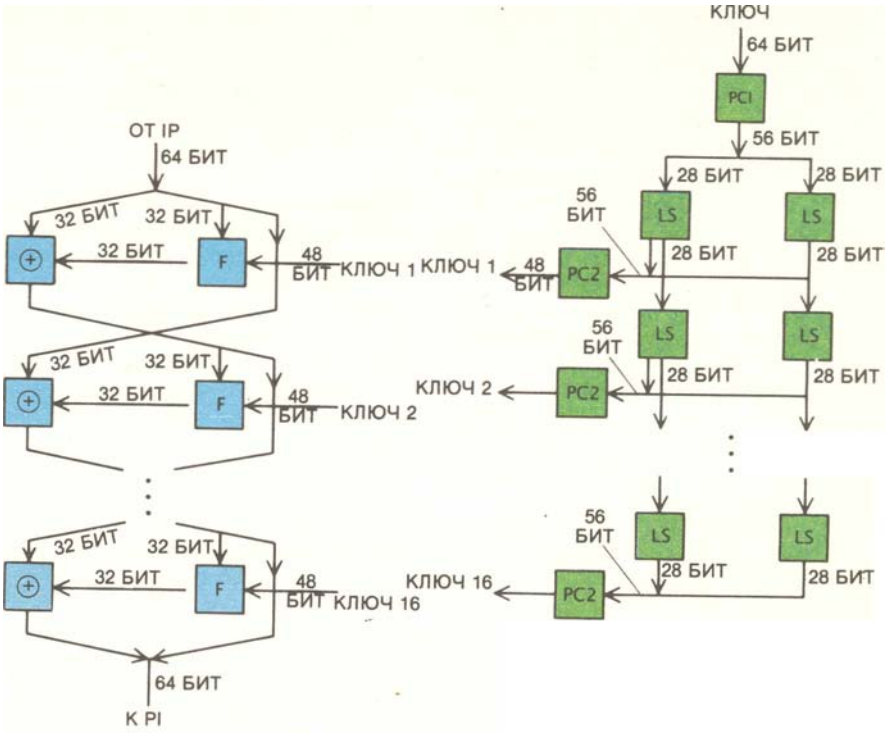
Заинтересованный посторонний может вторгнуться в систему общественных ключей, решив дискретную логарифмическую задачу за приемлемое время. Если число *y* в формуле

$$y = a^x (\mathrm{mod}\, p)$$

назвать дискретной степенью *a,* то *x* можно назвать дискретным логарифмом числау по основанию *a.* Дискретная логарифмическая задача заключается просто в том, чтобы найти *x* при известных *y* и *a.*

Существует ли метод более быстрый, чем простой перебор всех возможных значений *x* с возведением каждый раз в степень *x* числа *a* по модулю *p* и сравнением результата с числом *y?* Теоретики считают, что не существует, однако строго этого пока никто не доказал. Если дискретная логарифмическая задача в действительности окажется настолько же труднорешаемой, насколько это представляется, то очень большое значение *x,* скажем число с несколькими сотнями разрядов, должно уже отпугнуть любого любопытствующего вычислителя.

Диффи и Хеллман расширили свою схему рассылки обшественных ключей в криптосистему общественного пользования, придумав специальную функцию, названную ими односторонней функцией-люком. Вот как работает их система. Абонент **i** сети связи произвольно выбирает целое число k_i в качестве своего персонального ключа, а в справочнике публикует свой специальный кодируюший алгоритм E_i. Он также формулирует другой алгоритм D_i для декодирования,



Расширение версии центрального блока (слева) *и блока ключа-расписания* (справа)

который держит в секрете. Абонент *j* делает то же самое, публикуя свой кодирующий алгоритм E_j, но оставляя свой личный ключ k_j и декодирующий алгоритм D_j в секрете.

Позже, когда абонент *j* захочет послать сообшение абоненту i, он просто ознакомится с его кодирующим алгоритмом E_i по справочнику и воспользуется им для того, чтобы закодировать свое сообщение, исходную форму которого мы обозначим через m. Короче говоря, он передаст шифровку *y,* где

$$y = E_i(m).$$

Теперь абонент **i** без труда расшифрует сообшение с помощью своего секретного алгоритма D_i.

Эффективность такой системы шифров в основном зависела от того, удастся ли найти «односторонние кодирующие алгоритмы, т. е. такие кодирующие алгоритмы E_i, знание которых ничего не говорило бы о соответствуюших им декодирующих алгоритмах D_i. Идея подобной системы была очень привлекательной, хотя в то время, когда Диффи и Хеллман предложили ее, такие односторонние функции еше не были найдены.

Вот тогда-то и появилась криптосистема RSA, названная (по первым буквам фамилий) в честь Р.Ривеста, А.Шамира и Л.Адлемана, которые изобрели ее в 1978 г. Она и сейчас считается наиболее перспективной криптосистемой для сетей связи с известными ключами. Абоненты сети связи с шифром RSA выбирают каждый по два простых числа и кодирующий показатель степени. Например, абонент **i** выбирает простые числа P_i и q_i, а также кодируюший показатель степени e_i. Он публикует произведение $n_i = P_i \times q_i$ и свой кодирующий алгоритм, преобразующий ЧИсЛоВое сообщение *III* следуюшим образом:

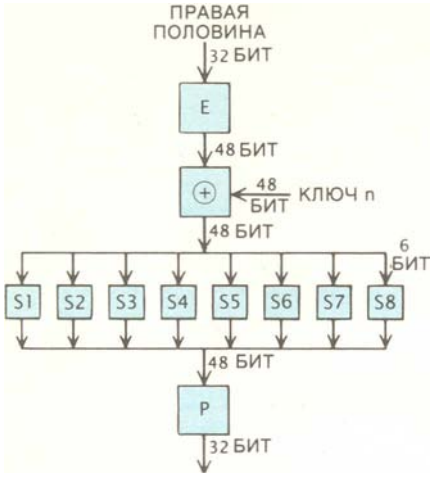
$$E_i(m) = m^{e_i} (\mathrm{mod}\, n_i).$$

Абонент *i* составляет также свой личный декодирующий алгоритм:

$$D_i(x) = x^{d_i} (\mathrm{mod}\, n_i).$$

Оказывается, что алгоритм D_i в точности противоположен алгоритму E_i. Эти два алгоритма применяются именно так, как предсказывали Диффи и Хеллман в своем проекте системы шифров для сетей связи. Как же это все работает? Каким образом абонент **i** находит число d_i?

Магическое число d_i нетрудно вычислить, зная числа e_i и $f_j = (P_i - 1) \times (q_i - 1)$. Простой алгоритм, изобретенный еще Евклидом, позволяет вычислить d_i как наибольший общий делитель чисел e_i и f_j. Это наибольшее число, на которое и e_i и f_i делятся без остатка. Теория чисел, многие положения которой кажутся несколько удивительными, гарантирует, что произведение d_i и e_i можно представить как сумму числа, КраТНого f_i, и единицы. Таким обра-



Расширенная версия модуля F

зом, когда абонент *i* лолучает сообщение, закодированное по его собственному алгоритму E_i, он фактически получает сообщение *m* в форме

$$m^{e_i} (\mathrm{mod}\, n_i).$$

Если он теперь возведет это выражение в степень d_i, то получит

$$m^{e_i \times d_i} (\mathrm{mod}\, n_i).$$

Новый показатель степени числа *m,* а именно e_i x d_i, как уже говорил ось, равен сумме некоторого КраТНого f_i и единицы. Однако, согласно другой теореме Евклида, число, возведенное в степень суммbI f_i и 1 по *модулю* n_i точности равно числу *m* и, следовательно, исходная форма сообщения обнажается в лучах математического света.

Вопрос о том, возможно ли расшифровать коды системы RSA, имеет приблизительно такой же ответ, который можно дать на аналогичный вопрос, касающийся системы с известными ключами. Если кто-нибудь изобретет по-настоящему быстрый алгоритм разложения на множители, который бы очень легко разбивал большие числа на простые составляющие, то шифры системы RSA будут без особого труда раскрыты. Увидев числоп$_i$ в общедоступном справочнике сети связи, нарушитель разобьет его на множителИ P_i и q_i и найдет число d_i, при ПОМОШИ которого сможет декодировать любое шифрованное сообщение от абонента *i*. Однако, по мнению теоретиков, такая задача разложения на множители, по-видимому, навсегда останется трднорешаемой.

Из-за недостатка места комментарии по поводу нашей статьи в сентябрьском номере я отложу до следующего раза.

Возвращение к ядерным реакторам

В США НАЗРЕВАЕТ сражение между сторонниками и противниками планируемой министерством энергетики постройки двух новых реакторов с целью получения сырья для ядерного оружия. За последние 15лет не поступало заказов на изготовление столь крупных ядерных реакторов ни гражданского, ни военного назначения.

По сообщению министерства энергетики, эти реакторы необходимы для гарантированного производства трития, изотопа водорода, получаемого путем радиоактивного облучения лития и используемого для увеличения мощности взрыва ядерного заряда боеголовки. Между тем скептики, такие как сенатор Э. Холлингс из шт. Южная Каролина, задаются вопросом, оправданы ли затраты на постройку этих двух реакторов, оцениваемых в 6,8 млрд. долл., даже при самых пессимистичных прогнозах относительно возможных сокращений вооружений в будущем?

Многие признают необходимость некоторого увеличения производства трития. Ежегодно 5,5% трития в боеголовках распадается и требуется его пополнение. Существующие же реакторы, производящие тритий, конструктивно отстают от современных в требований по обеспечению безопасности и они устарели из-за длительной эксплуатации.

На единственной атомной электростанции министерства энергетики Savannah River Plant в шт. Южная Каролина, где про изводится тритий, в прошлом году из-за образовавшихся в корпусе трещин был списан один старый реактор на тяжелой воде; три других реактора на той же станции, являющиеся в настоящее время единственным источником трития в стране, часто приходится останавливаТЬ для проведения ремонтных и регламентных работ. По соображениям безопасности, эксплуатация этих реакторов ОСУществляется в половину их мощности.

Результаты специального исследования, проведенного в прошлом году Национальным научно-исследовательским советом, вызывают серьезные сомнения в отношении возможности дальнейшей эксплуатации этих реакторов. В материалах исследования указывается на наличие в охладительной системе реакторов трещин, образовавшихся в результате напряжений и коррозии, отмечается «отсутствие твердой уверенности» в сПособности защитной оболочки реакторов предотвратить утечку радиоактивности в случае крупной аварии и критикуется руководство этим объектом министерством энергетики. Уверенность в надежности реакторов была еще более по колеблена в августе прошлого года, когда инспекторы по технике безопасности оказались свидетелями того, как безуспешно пыта·· ясь запустить один из реакторов, обслуживающий персонал станции вдруг стал повышать нагрузку, вместо того чтобы немедленно заглушить его. Несколько дней спустя, на этом же реакторе был отмечен самопроизвольный «всплеск» нагрузки; возможно, это и не было связано с предыдущей неполадкой, но тем не менее не могло не насторожить.

О планах постройки двух новых реакторов для производства трития министерство энергетики объявило незадолго до этого случая. Один большой реактор на тяжелой воде намечено было в кратчайший срок построить на территории Savannah River Plant; его сооружение планировалось произвести за срок около 10 лет. Одновременно предполагалось приступить к проектированию так называемой модульной высокотемпературной ядерной установки с газовым охлаждением и затем соорудить ее на территории Национальной техиической лаборатории в шт. Айдахо. Как указывает министерство энергетики, одного реактора на атомной станции Savannah River Plant вполне хватило бы на удовлетворение потребности в тритии, а реактор в шт. Айдахо дополнительно производил бы еще 50% трития сверх необходимого количества.

Дополнительное количество трития может быть получено, даже если не удастся достичь новых соглашений с СССР о крупномасштабных сокращениях вооружений. Каждое такое соглашение заметно уменьшает потребность в тритии и резко увеличивает его наличное количество за счет изъятия из демонтированных боеголовок. Как считает Д. Олбрайт из Ассоциации американских ученых, решение о постройке даже одного дополнительного реактора вполне можно было бы отложить до того, как станут известны результаты проходящих сейчас переговоров о сокращении стратегических вооружений. Сенатор от шт. Массачусетс Э. Кеннеди, как и ряд других, в чрезмерном расширении мощностей по производству трития усматривает опасность для будущих попыток наладить взаимный контроль за